UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/693,517	10/19/2000	Lawrence A. Crowl	SUN1P381/P4502	7922

22434 7590 11/21/2005

BEYER WEAVER & THOMAS LLP
P.O. BOX 70250
OAKLAND, CA 94612-0250

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

DATE MAILED: 11/21/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/693,517	CROWL ET AL.	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 October 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1 and 3-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1 and 3-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the application filed 10/07/2005. As indicated by Applicants, claims 1, 3-4, 8, 10, 14-15, and 17-19 are amended and claim 2 canceled.

Claims 1, and 3-20 have been submitted for examination.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

3. Claims 10-13 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

The phrase 'suitable for' (line 2, 8) does not enable a reasonable construction that a claimed entity (e.g. a compiler) is actually performing an operation or being virtually idle/non-functional. This indefiniteness will be treated as if the compiler is compiling source code.

Dependent claims 11-13 are also rejected for failing to remedy to the above deficiency.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person

having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1, 3-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Burch, USPN: 6,308,320 (hereinafter Burch), in view of Fitzgerald, USPN: 5,408,665(hereinafter Fitzgerald).

As per claim 1, Burch discloses a method of compilation of source program into a object file using one or more associated depository of instances, wherein each instance is a specialization of a template and its available operations (e.g. *object files* - col. 7, lines 15-17, 35-36 - Note: object file being created via a compiling process reads on instance being created a grammar-based parser -- operating on predetermined symbol tree of a given grammar-- reads on instance of specializing a template -- such as a parsing tree -- instantiated for a particular source code constructs), the method comprising:

identifying one or more instances available for use in the depository of instances (e.g. *reuse depository 208* – col. 10, lines 26-45; col. 11, lines 32-45) using linker symbol names (e.g. col. 10, lines 34-45) for the instance(s);

receiving a first request to create a first instance of said object file (e.g. col. 11, lines 32-45 – Note: attempt to see if new instance of object file is to be created reads on get a request to do so); and

determine whether the first instance has been identified in the one or more depository (e.g. col. 11, lines 32-45 – Note: checking if the object file instance is already in a depository reads on whether a first instance has been identified in such depository);

and when the first instance has not been identified in one of such depository, creating the first instance (e.g. col. 11, lines 60-63); and when the first instance has been identified in one of

such depository using the linker symbol name (Note: a linker inherently operates with linker symbol name) of the instance as a reference to the instance already in the depository (e.g. *passed to the linker*- col. 11, lines 40-47; Fig. 6B – Note: not creating when it said instance already exists would have been implicitly disclosed), thereby avoiding duplication and reducing the time needed for compilation (Note: the limitation does not add more teaching to the upper part of the claim whereby the avoiding of duplicate is inferred).

But Burch does not specify that the depository to store an instance is library of instances. However, the purpose of library storing object files to be reused at different compilation instances was a known concept at the time the invention was made; and Burch has shown that libraries can be used dynamically with the previously stored object files at link time for creating additional or expanded object files as a result of combining reusable libraries and reusable of object files storage (e.g. col. 3, lines 55-64; libraries 114 – Fig. 1; col. 8, lines 39-42; col. 9, lines 38-47; col. 9, line 67 to col. 10, line 3). So the concept of keeping object files in a repository for dynamic linking of objects and the library of files reusable for linking would be equivalent to each other; hence Burch has disclosed a form of library object files via the depository. In the hypothetical case that this concept of library object file in view of Burch's reusable store would not be clear and self-evident, this concept would have been obvious.

Analogous to Burch's approach in alleviating compiler linking resources, Fitzgerald in a method to compile and link compiled code into executable discloses the use of one or more libraries for scanning in order to identify which instances are needed for the final linking of the executable subsequent linking thus to obviate useless re-instantiation resources (e.g. *contained in library files, marked ... needed* – col. 9, line 32 to col. 10, line 8; Fig. 4A; Fig. 3A-C). Based on

the concept of persisting library files needed for dynamic reuse and object linking by Burch as recognized further via the repository of object files for with reuse thereof, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the depository of reusable object files by Burch (*reuse depository*) so it be library of object instances as taught by Fitzgerald because in conjunction with alleviating resources for having to create new object files as taught by Burch, the use of library object files of object files would enable dynamically, fast and selectively loading/activating object instances when resolving linking references (as originally intended by Burch) as evidenced by Fitzgerald, thus provide a more efficient process in which recreating of pre-stored object codes can be obviated and dynamically reused in the linking process thereby would expedite the final code linking as evidenced by the tight, fast and immediate association of library files during linking so well known in compiler technologies.

As per claim 3, Burch discloses

creating the first instance when such linker symbol name is identified as not matching any available instances in the one or more libraries(e.g. col. 11, lines 32-45).

Although Burch does not explicitly specify that the creation of instance when the linker symbol name does not match the identified symbol names for instance available for use in the one or more libraries, the symbol name leading (col. 10, lines 34-45; col. 11, lines 32-45) to searching of the appropriate object file in the depository and the subsequent creation in case of no match implicitly teaches the above limitation.

As per claim 4, Burch discloses accessing one or more libraries and selecting object file name referenced by linker symbol names based on what files are available in the library and

saving the names (e.g. Fig. 7A, B; re cited parts of claim 1, 3 – Note: identify names that already exist implicitly disclose saving of names). As for the limitation as to selecting linker symbol names that correspond to instances for use in one of the libraries, in view of the process to identify object code instance to create anew or use without recreating as addressed in claim 1 using the symbol referencing therein by Burch and the rationale as to implement a library of object to be support linking process by Fitzgerald, the above limitation would also have been obvious for the same reasons as set forth in claim 1.

As per claim 5, Fitzgerald discloses examining and extracting linker symbol names likely to correspond to instances of library object files code (e.g. Fig. 4A, 4C; 5A); and the rationale as to using the library and the selective marking of instances by Fitzgerald combined with the selective creation of object instances by Burch would have made the current limitation obvious for the same benefits as mentioned in claim 1.

As per claim 6, Burch (in combination with Fitzgerald) disclose a linker symbol names that include predetermined sequence of characters (see Burch: *file name* - Fig. 7B, C, D; see Fitzgerald: Fig. 4A, 4C; 5A)

As per claim 7, Burch disclose hash table (Fig. 7B, C, D).

As per claim 8, Burch (combined wiht Fitzgerald's teachings) discloses obtaining a first linker symbol name for the first instance(e.g. col. 10, lines 34-45);

comparing the symbol name with those selected linker symbol names likely to correspond to object instances (e.g. Fig. 7A-C), and

creating the first instance when the first linker symbol does not match any of those selected linker symbol names likely to correspond to said instances (col. 11, lines 32-45).

But Burch does not teach explicitly teach template instance even though Burch code is reusable code with intermediate form (Fig. 3A) similar to object-oriented language and teaches a hash directory to be overridden (fig. 6B). But in view of the instance of object file created by instance of running a compiler working with a predetermined hierarchy of symbol and grammar tree structure for a specific sets of source code constructs, the instance of template or object file being specialized by such parsing process with its internal parsing operations is disclosed.

As per claim 9, Burch discloses intermediate code (Fig. 3) and Fitzgerald discloses source program in C++ (e.g. col. 4, lines 58-63). Object oriented code like C++ or Java being portable through intermediate or bytecodes was a well known concept and at the time the invention was made, implementing code with intermediate form so that it is reusable and object oriented as suggested by Burch in view of Fitzgerald C++ teaching would have been obvious for portability benefits and all pertinent advantages in OO programming language.

As per claim 10, Burch discloses a system suitable for compilation of source program into object files, the system comprising:

a source program (e.g. Fig. 3);

a depository including at least one instance available for use by the program, the instance being a specialization of a template and its available operations (e.g. *reuse depository 208* – col. 10, lines 26-45; *object files* - col. 7, lines 15-17, 35-36) an instance identifiable by a linker symbol name (col. 10, lines 34-45; Fig. 7B);

an enhanced compiler suitable for compilation of source code (Fig. 3), such compiler accessing the depository object file to identify the one instance available in the depository (e.g. col. 11, lines 32-45) by the linker symbol name (Note: a linker inherently operates with linker

symbol name) of the instance (e.g. *passed to the linker*- col. 11, lines 40-47; Fig. 6B – Note: not creating when it said instance already exists would have been implicitly disclosed), the compiler thereby avoiding duplication and reducing the time needed for compilation (Note: the limitation does not add more teaching to the upper part of the claim whereby the avoiding of duplicate is inferred)

But Burch does not explicitly specify that the depository to create a first instance or instance of object file is one or more libraries thereof. This library limitation, however, has been addressed in claim 1 above, hence is rejected herein using the same rationale set forth therein.

As per claims 11 and 12, Burch (in combination with Burch) discloses retrieving, i.e. using an instance extractor , an instance of the library, available to be use for the program generating; and comparing one instance available with a desired instance (e.g. col. 11, lines 32-45).

As per claim 13, Fitzgerald discloses storage of an instance name (e.g. e.g. Fig. 7A, B; re cited parts of claim 1, 3 – Note: identify names that already exist implicitly disclose saving of names).

As per claim 14, Burch discloses a method of compilation into a object file using one or more associated depository with instances available for use by the program, wherein each instance is a specialization of a template and its available operations (e.g. *object files* - col. 7, lines 15-17, 35-36), the method comprising:

examining a linker symbol name of one or more associated depository (e.g. col. 10, lines 34-45; Fig. 5);

extracting from the linker name one or more linker symbol names that are likely to correspond to the stored linker symbol names (e.g. Fig. 5, 7A-D – Note: a directory of file pathname being passed to a linker – see col. 12, lines 14-20 reads on a linker name table because a linker necessarily includes a internal table for effecting the resolving of symbols being passed);

storing the extracted symbol names (e.g. Fig. 5, 7A-D);

receiving a first request to create a first instance, said first instance during compilation, said instance having a first linker symbol name (col. 11, lines 32-45);

comparing the first linker symbol name with one or more stored linker symbol names; and creating the first instance when such symbol name is not yet stored (e.g. e.g. col. 11, lines 60-63; col. 11, lines 32-45 – Note: checking if the object file name instance is already in a depository reads on comparing based on whether a first instance has been identified in such depository); thereby avoiding duplication and reducing the time needed for compilation.

But Burch does not specify that the storage for depository of object files is one or more libraries of instances nor does Buch explicitly disclose examining a linker name table. This library and linker table limitations in view of the teaching by Fitzgerald using linking tables (e.g. Fig. 4A, 5A, 6A), however, has been addressed in claim 1 above, hence is rejected herein using the same rationale set forth therein.

As per claim 15, Burch (in combination with Fitzgerald) discloses a comparing without transforming the names in the one or more libraries (e.g. Fig. 7A-D).

As per claim 16, refer to rejection of claims 7 and 9.

As per claim 17, Fitzgerald discloses a computer-readable medium to include code for implementing a method with the limitations as recited in claim 1, namely the steps of

identifying(instances), receiving (request), determining(instance available), creating (first instance). Hence the instant claim step limitations are herein rejected (using Fitzgerald in combination with Burch) with the corresponding rejections as set forth in claim 1.

As per claims 18-20, these are computer medium claims corresponding to claims 3, 4, and 6 above, respectively, hence are rejected herein using the rejection set forth therein.

Response to Arguments

6. Applicants' arguments filed 10/07/2005 regarding claims 1, and 3-20 have been considered but are not persuasive. Following are the Examiner's observations in regard thereto.

(A) Applicants have submitted that while the 'invention mechanism is to avoid producing multiple instances across object files and libraries', Burch 'reduces the number of compilation by checking if a object file already exists in the repository' and Burch is not operating to 'reduce ...work ... in each compilation by checking if an instance exists in one or more library object files' (Appl. Rmrks, pg. 9, top 2 para). The claim does not contain a recital of 'reduce the amount of work in each compilation by checking if an instance exists in one or more library object files'; hence the argument seems to be raising the patentability of a limitation that clearly does not exist. Besides, the rejection has pointed out what in Burch has been analogized as what is recited as 'instance'. From the claim and the specifications, an instance is a specialized instance of some programming language-related template. And for one skill in the art a template is a reusable (and initially empty) container having structure that can be primarily populated (i.e. instantiated) with a plurality of data like externally inputted variables, language constructs, function code or token/text symbols; and a template signifies the instantiation of an otherwise empty object with the data filling as set forth above. A parsing process wherein source code

constructs following a parsing and lexical scheme based on a set of grammar rules reads on a template being the parsing tree as an example. And if a object code is the result of a process by which code token/symbols are parsed and converted (i.e. using parse tree template) into a object module/file based on some particular language, such object file is an instance being a specialization of the above-mentioned template. Burch's object file being persisted reads on instance in a repository. Hence, the argument is non persuasive for lack of clearly defining 'instance' and 'template' in terms so as to undeniably distinguish over an object file as analyzed from above.

(B) Applicants have submitted that the invention of claim 1 avoids 'duplicate instances', 'not duplicate identical realizations'; and that claim 1 does not describe 'extraction of objects' but concerns with 'recognition of an instance' (Appl. Rmrks, pg. 9, 3rd para). The language used to recite claim 1 does not provide any explicit teaching --as perceived by the Office Action-- in terms that would enforce the allegation such as 'avoiding identical realizations', nor does it enforce 'recognition of an instance' so make it distinguishable from Burch's scanning method to determine whether instances being previously stored already exist, particularly because of the analysis as set forth in section A above.

(C) Applicants have submitted that the amended claim 1 with its specific limitations are not taught by Burch and Fitzgerald, alone or in combination (Appl. Rmrks, pg. 10, top). The argument is founded on the fact of better defining 'instance'; but as mentioned above, Burch has disclosed 'instance'; and Applicants by mere repeating the claimed feature, necessarily fails to point out exactly where Burch's cited parts do not meet the claimed limitations as set forth in the Office Action.

(D) Applicants have submitted that Burch, Fitzgerald with Perks in the rationale used in the rejection do not establish prima facie. The rejection as set forth herein has rendered this Applicants' arguments moot. The rest of the arguments (Appl. Rmrks, pg. 10), depend on the validity of Applicants' ground of rebuttal as proffered above in section A and B. These are not convincing for the same reasons as set forth in section A or B.

For the above reasons, the claims stand rejected as set forth in the Office Action.

Conclusion

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
November 7, 2005

TODD INGBERG
PRIMARY EXAMINER